OCP U.S. SUMMIT 2016 March 9-10 | San Jose, CA

OCP U.S. SUMMIT 2016

Towards a Firmware Update Standard

Mallik Bulusu
UEFI Dev. Lead
Microsoft Corporation
mallikb@microsoft.com

Vincent Zimmer
Sr. Principal Engineer
Intel Corporation
vincent.zimmer@intel.com

What Did We Say in 2015?

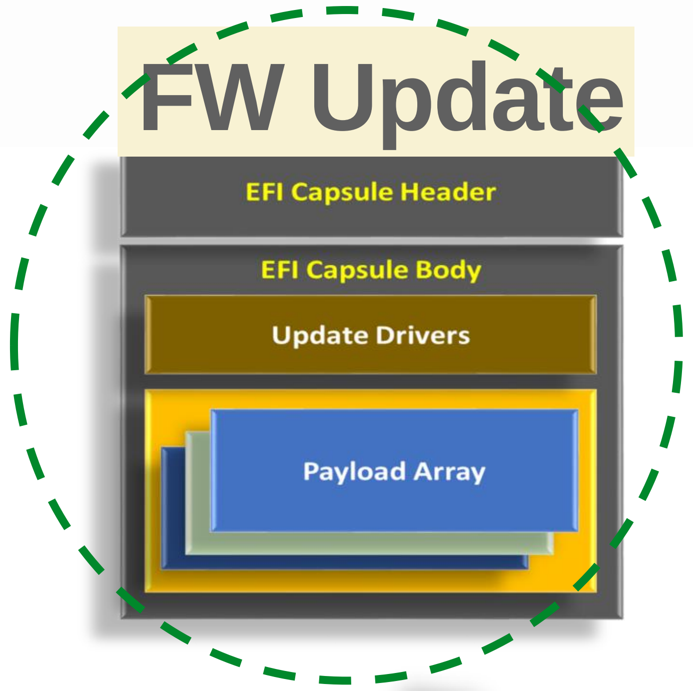
Cloud scale offers unique challenges to development of firmware

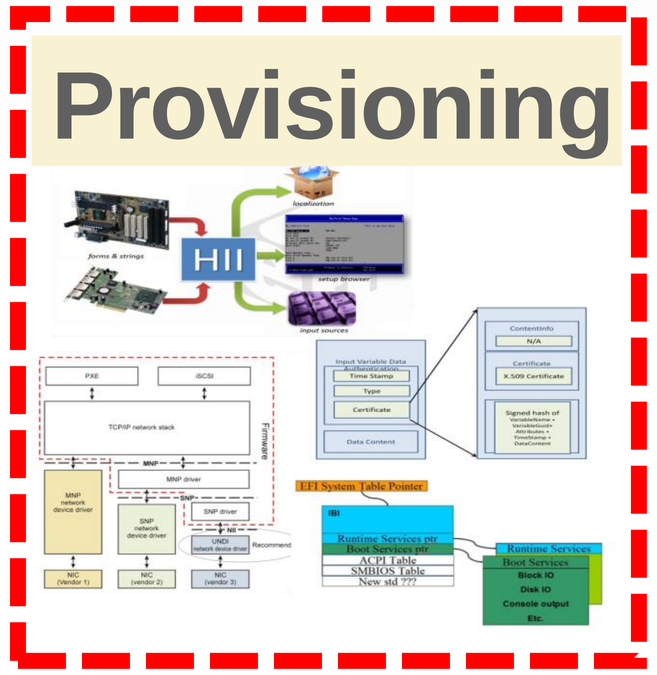
Factors affecting FW updates –

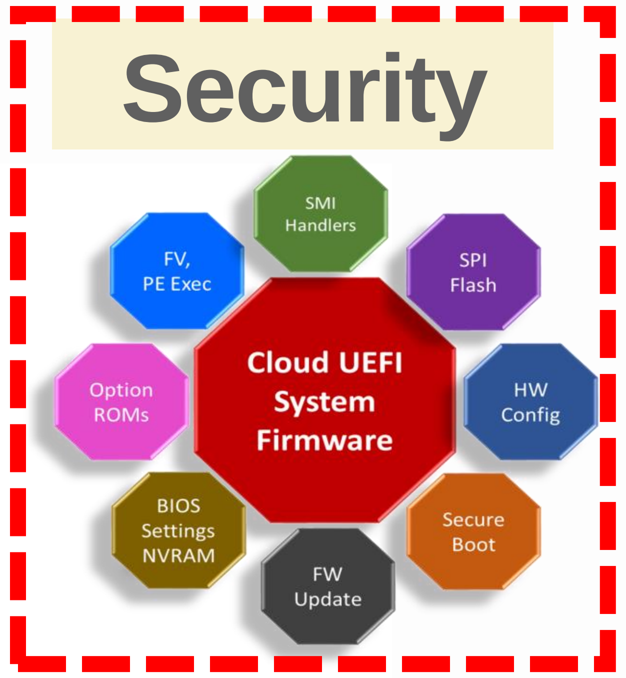
- Different Types
- Deployment
- Conformance
- Availability
- Recovery

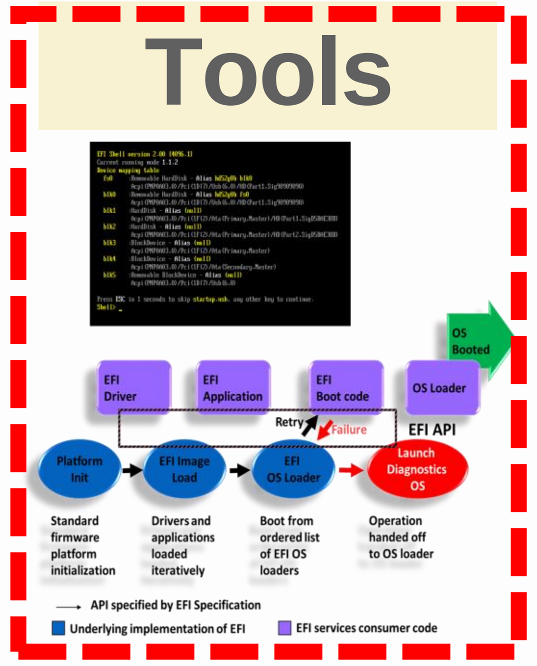
We can't avoid FW updates –

- Bug Fixes
- Performance Improvements
- Security









```
EFI Shell version 2.00 [4096.1]
Current running mode 1.1.2
Device mapping table
  fs0   :Removable HardDisk - Alias hd52g0b blk0
  blk0  :Removable HardDisk - Alias hd52g0b fs0
  blk1  :HardDisk - Alias (null)
  blk2  :HardDisk - Alias (null)
  blk3  :BlockDevice - Alias (null)
  blk4  :BlockDevice - Alias (null)
  blk5  :Removable BlockDevice - Alias (null)

Press ESC in 1 seconds to skip startup.nsh, any other key to continue.
Shell> _
```

OS Booted
EFI Driver
EFI Application
EFI Boot code
OS Loader
Retry
Failure
EFI API
Platform Init
EFI Image Load
EFI OS Loader
Launch Diagnostics OS
Standard firmware platform initialization
Drivers and applications loaded iteratively
Boot from ordered list of EFI OS loaders
Operation handed off to OS loader
API specified by EFI Specification
Underlying implementation of EFI
EFI services consumer code

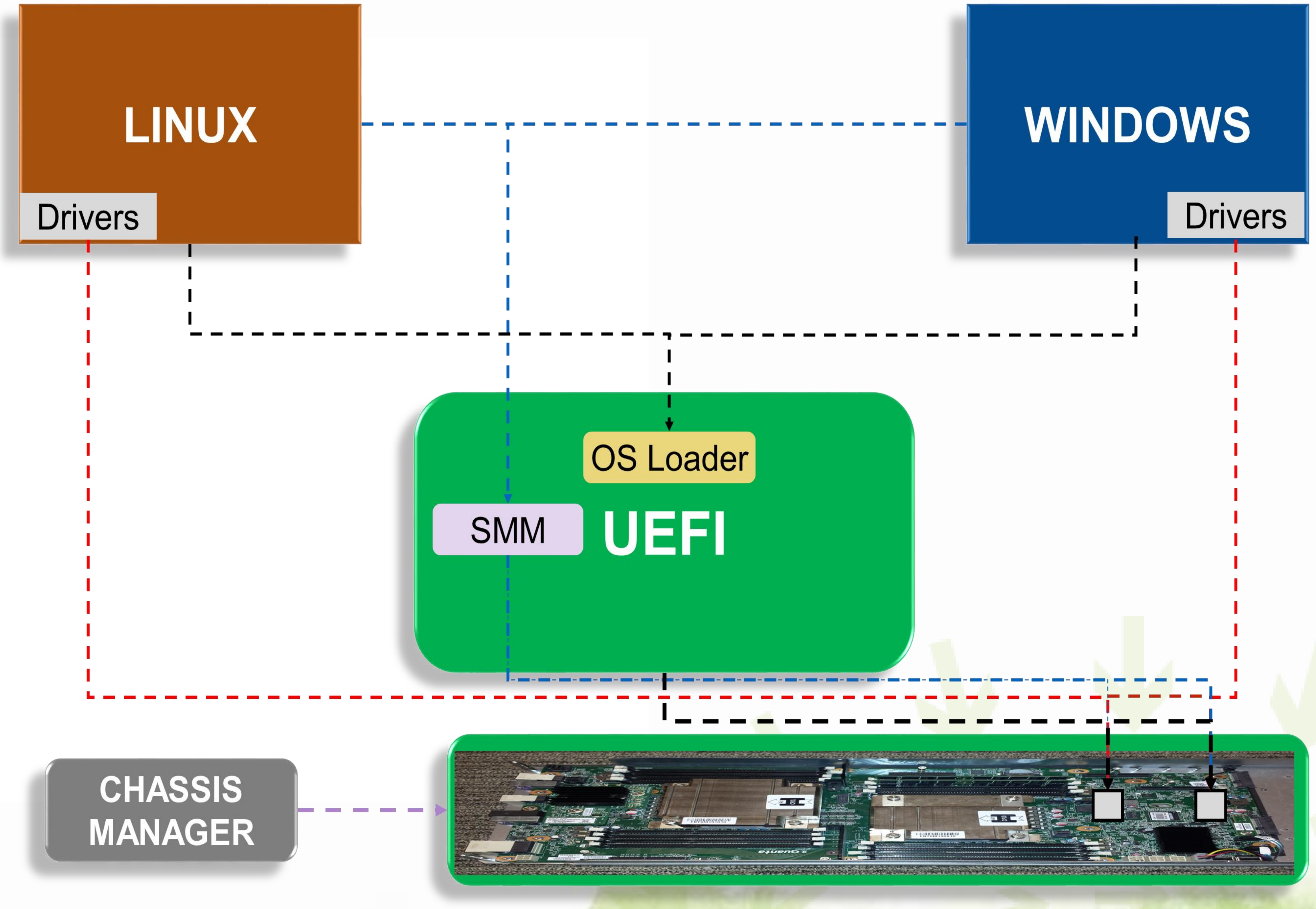
Firmware Update Challenges

- Components from multiples vendors
- Delivering firmware
- Different types of devices
- Recovery from failures
- Node equivalence across datacenter
- Security, security, security……

ARM
CPLD
NIC
UEFI
UCODE
STORAGE
BMC
MISC.
FPGA
ME

FW Update Scenarios in Cloud

Mode	Advantages	Opens
UEFI	API / Envelope definition; System/ Device Firmware story;	System Resets;
SMM	Silent	Device Firmware story. Few updates take affect after reset.
OS Driver	Silent	Standardization (?)
OOB	Controlled Environment	Arbitration with host context; Bandwidth



NOMENCLATURE:

System Firmware: Elements that **are required** for system boot, for e.g. BIOS, ME, BMC.

Device Firmware: Elements that **are not required** for system boot, for e.g. OPROMs, etc.

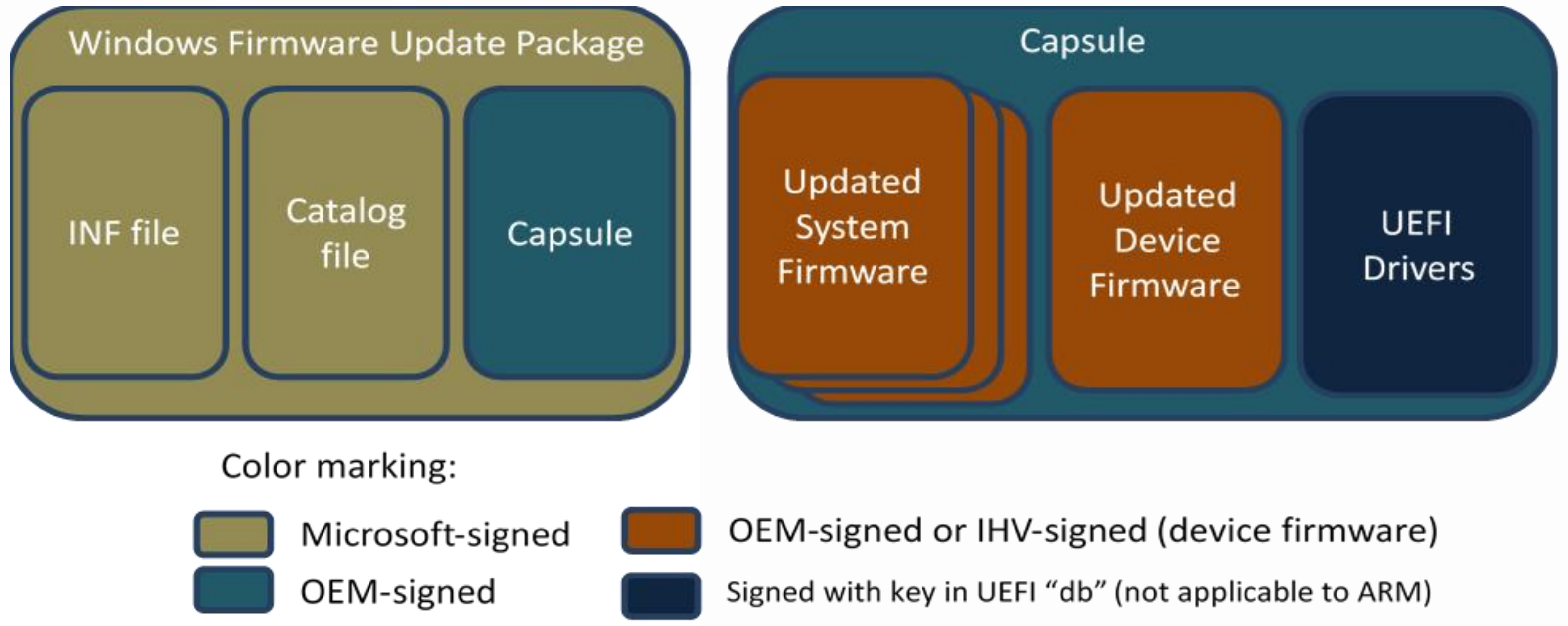
What is a Capsule?

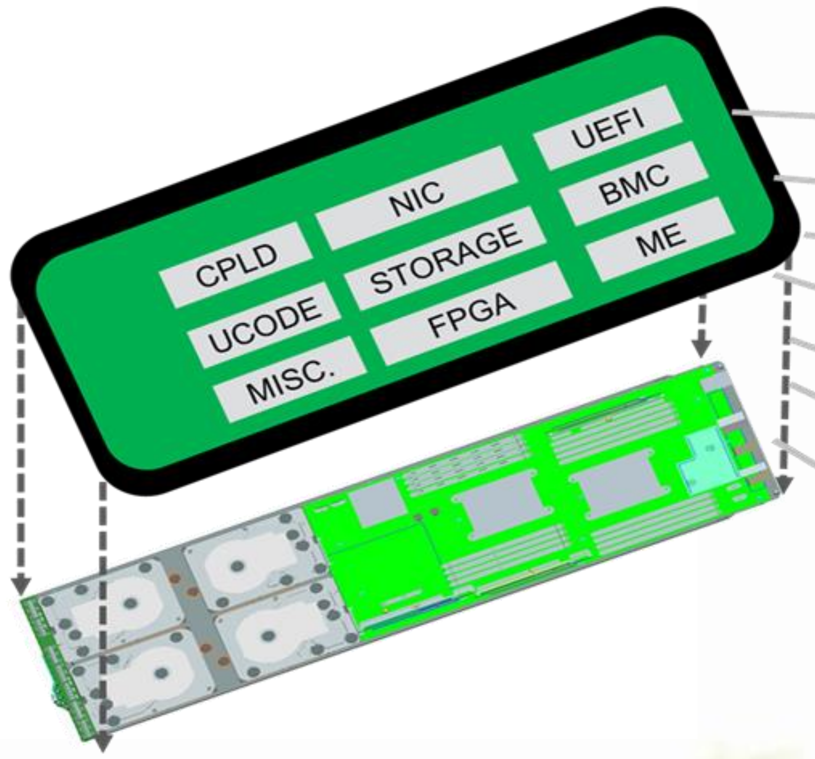
- **API**
  - System or Device initialization
  - Carries interfaces to interact with Device
  - Presents interfaces to Read, Modify, Verify Firmware
- **Envelope**
  - Parsing the blob
  - Integrity checks
  - Validity checks
  - Update mode independent

Prevent rogue FW updates

Provisions in UEFI

- Publish metadata (ESRT)
  - Firmware Resource Description
  - Published by BIOS
  - Firmware update status
- Capsules on Disk
- Smart capsules (FMP)
  - Published by driver
  - Capsule comprises of header & body
  - Capsule body comprises
    - EFI Firmware Management Capsule Header
    - Optional Drivers
    - Payloads
  - Updates handled during pre-boot





FW	GUID
BIOS	{ GUID$_1$}
BMC	{ GUID$_2$}
ME	{ GUID$_3$}
NIC	{ GUID$_4$}
Storage FW	{ GUID$_5$}
UCODE	{ GUID$_6$}
CPLD	{ GUID$_7$}
Etc..	{ GUID$_{..}$}

Example ESRT

Can we extend these provisions to other Firmware Update scenarios?

Envelope is More Universal

- The capsules itself has a heade and body
- Capsule can be firmware volume with encapsulation sections
- Sections can include compression, signing, encryption
- Some well-known section types in the UEFI PI specification
- Can be vendor specific

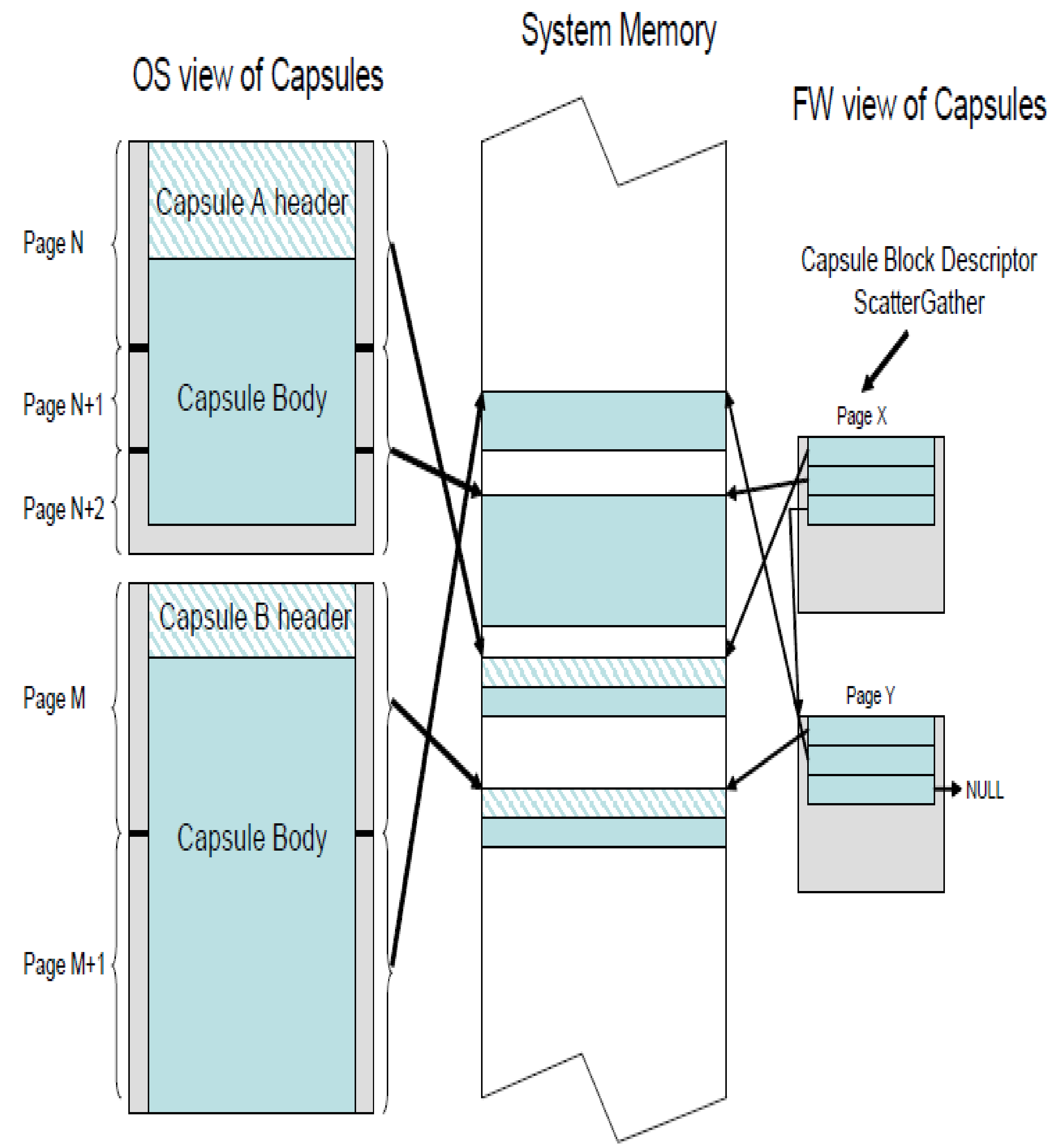


Figure 22. Scatter-Gather List of EFI_CAPSULE_BLOCK_DESCRIPTOR Structures

How to Use the Envelope?

- **In-band** – EFI UpdateCaspule or Capsule-on-Disk
- **Out-of-Band** –via Service Processor
- **OS** – Linux FW API, MS .cab etc.

- Capsules are GUID based
  - Some well known GUIDs
  - Others can be
    - Vendor specific;
    - Standards group specific (think OCP?)

Security

- Envelope to also include signing.
- Already have UEFI Secure Boot for UEFI executables (apps & drivers)
- Have UEFI envelope possibilities.
  - Security Version Numbers (SVN’s)
  - golden images
  - roll-back concerns

Questions?

- Security (see before)
- Should in band (e.g., UpdateCaspule) and out of band (OOB) be harmonized?
- Same binary to OS driver for device?
- How to get inventory for node equivalence?
- Share tools?
- Namespace of GUID’s for OCP style gear?

What type of information that could appear in an OCP spec on ‘updates?’ What can land in ’16 given aforementioned ecosystem readiness?

More information

http://www.opencompute.org – OCP specs

http://www.uefi.org – UEFI, ACPI, Shell, PI Specifications

http://www.Tianocore.org – open source UEFI

http://firmware.intel.com – white papers, training

https://www.dmtf.org/sites/default/files/UEFI-DMTFWorkReg1_2_v2.pdf - DMTM & UEFI

https://svn.code.sf.net/p/edk2/code/trunk/edk2/IntelFrameworkModulePkg/Library/DxeCapsuleLib – Capsule Update Implementation

https://firmware.intel.com/blog/uefi-and-cloud – UEFI & Cloud discussion at UEFI Plugfest

https://blogs.intel.com/evangelists/2015/06/23/better-firmware-updates-in-linux-using-uefi-capsules/ – Community Overview

https://msdn.microsoft.com/en-us/library/windows/hardware/dn917887(v=vs.85).aspx – Microsoft Windows Firmware Updates

https://blog.uncooperative.org/blog/2015/09/16/an-update-on-firmware-updates/ – Linux Firmware Updates

OPEN

Compute Project